```
PPPPPPPPPPPP        AAAAAAAAA     TTTTTTTTTTTTTTT     CCCCCCCCCCC  HHH         HHH
PPPPPPPPPPPP        AAAAAAAAA     TTTTTTTTTTTTTTT     CCCCCCCCCCC  HHH         HHH
PPPPPPPPPPPP        AAAAAAAAA     TTTTTTTTTTTTTTT     CCCCCCCCCCC  HHH         HHH
PPP         PPP   AAA         AAA       TTT         CCC           HHH         HHH
PPP         PPP   AAA         AAA       TTT         CCC           HHH         HHH
PPP         PPP   AAA         AAA       TTT         CCC           HHH         HHH
PPP         PPP   AAA         AAA       TTT         CCC           HHH         HHH
PPP         PPP   AAA         AAA       TTT         CCC           HHH         HHH
PPP         PPP   AAA         AAA       TTT         CCC           HHH         HHH
PPPPPPPPPPPP      AAA         AAA       TTT         CCC           HHHHHHHHHHHHHHH
PPPPPPPPPPPP      AAA         AAA       TTT         CCC           HHHHHHHHHHHHHHH
PPPPPPPPPPPP      AAA         AAA       TTT         CCC           HHHHHHHHHHHHHHH
PPP               AAAAAAAAAAAAAAA       TTT         CCC           HHH         HHH
PPP               AAAAAAAAAAAAAAA       TTT         CCC           HHH         HHH
PPP               AAAAAAAAAAAAAAA       TTT         CCC           HHH         HHH
PPP               AAA         AAA       TTT         CCC           HHH         HHH
PPP               AAA         AAA       TTT         CCC           HHH         HHH
PPP               AAA         AAA       TTT         CCC           HHH         HHH
PPP               AAA         AAA       TTT         CCC           HHH         HHH
PPP               AAA         AAA       TTT            CCCCCCCCCCC  HHH         HHH
PPP               AAA         AAA       TTT            CCCCCCCCCCC  HHH         HHH
PPP               AAA         AAA       TTT            CCCCCCCCCCC  HHH         HHH
```

FILEID**PATSYM

```
PPPPPPPP      AAAAAA    TTTTTTTTTT    SSSSSSSS  YY      YY  MM      MM
PPPPPPPP      AAAAAA    TTTTTTTTTT    SSSSSSSS  YY      YY  MM      MM
PP      PP   AA    AA      TT       SS          YY      YY  MMMM  MMMM
PP      PP   AA    AA      TT       SS          YY      YY  MMMM  MMMM
PP      PP   AA    AA      TT       SS            YY  YY    MM  MM  MM
PP      PP   AA    AA      TT       SS            YY  YY    MM  MM  MM
PPPPPPPP     AA    AA      TT         SSSSSS        YY      MM      MM
PPPPPPPP     AA    AA      TT         SSSSSS        YY      MM      MM
PP           AAAAAAAAAA    TT               SS      YY      MM      MM
PP           AAAAAAAAAA    TT               SS      YY      MM      MM
PP           AA    AA      TT               SS      YY      MM      MM      ....
PP           AA    AA      TT               SS      YY      MM      MM      ....
PP           AA    AA      TT       SSSSSSSS        YY      MM      MM      ....
PP           AA    AA      TT       SSSSSSSS        YY      MM      MM      ....

LL             IIIIII      SSSSSSSS
LL             IIIIII      SSSSSSSS
LL               II      SS
LL               II      SS
LL               II      SS
LL               II        SSSSSS
LL               II        SSSSSS
LL               II              SS
LL               II              SS
LL               II              SS
LL               II              SS
LLLLLLLLLL     IIIIII    SSSSSSSS
LLLLLLLLLL     IIIIII    SSSSSSSS
```

```
;     1          0001  0 MODULE PATSYM (
;     2        L 0002  0                 %IF %VARIANT EQL 1
;     3          0003  0                 %THEN
;     4          0004  0                         ADDRESSING_MODE (EXTERNAL = LONG_RELATIVE, NONEXTERNAL = LONG_RELATIVE),
;     5          0005  0                 %FI
;     6          0006  0                 IDENT = 'V04-000') =
;     7          0007  1 BEGIN
;     8          0008  1
;     9          0009  1 !
;    10          0010  1 !*****************************************************************************
;    11          0011  1 !*                                                                           *
;    12          0012  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
;    13          0013  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                    *
;    14          0014  1 !*  ALL RIGHTS RESERVED.                                                      *
;    15          0015  1 !*                                                                           *
;    16          0016  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED     *
;    17          0017  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
;    18          0018  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER     *
;    19          0019  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY     *
;    20          0020  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY     *
;    21          0021  1 !*  TRANSFERRED.                                                              *
;    22          0022  1 !*                                                                           *
;    23          0023  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE     *
;    24          0024  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT     *
;    25          0025  1 !*  CORPORATION.                                                              *
;    26          0026  1 !*                                                                           *
;    27          0027  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS     *
;    28          0028  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                   *
;    29          0029  1 !*                                                                           *
;    30          0030  1 !*                                                                           *
;    31          0031  1 !*****************************************************************************
;    32          0032  1 !
;    33          0033  1
;    34          0034  1 !++
;    35          0035  1 ! FACILITY:     PATCH
;    36          0036  1 !
;    37          0037  1 ! ABSTRACT:
;    38          0038  1 !
;    39          0039  1 !       This module contains the routines for manipulating the
;    40          0040  1 !       user-defined and global symbols.
;    41          0041  1 !
;    42          0042  1 ! ENVIRONMENT:  STARLET, user mode, interrupts disabled.
;    43          0043  1 !
;    44          0044  1 ! Version:      V02-009
;    45          0045  1 !
;    46          0046  1 ! History:
;    47          0047  1 !       Author:
;    48          0048  1 !               Carol Peters, 13 Dec 1976: Version 01
;    49          0049  1 !
;    50          0050  1 ! MODIFIED BY:
;    51          0051  1 !
;    52          0052  1 !       V02-009 PCG0001         Peter George            02-FEB-1981
;    53          0053  1 !               Add require statement for LIB$:PATDEF.REQ
;    54          0054  1 !
;    55          0055  1 !       Modified by:
;    56          0056  1 !               Kathleen Morse, 19 Oct 1977: Version X01.00
;    57          0057  1 !
```

```
;    58          0058  1 ! Revision history:
;    59          0059  1 !
;    60          0060  1 !  NO   DATE            PROGRAMMER          PURPOSE
;    61          0061  1 !  --   ----            ----------          -------
;    62          0062  1 !
;    63          0063  1 !  00   19-OCT-77        K.D. MORSE          ADAPT VERSION 15 FOR PATCH
;    64          0064  1 !  01   5-JAN-78         K.D. MORSE          NO CHANGES FOR 16-17.
;    65          0065  1 !  02   7-MAR-78         K.D. MORSE          ADD PAT$VAL_TO_DEF.
;    66          0066  1 !  03   7-APR-78         K.D. MORSE          ADD PAT$FIND_VAL TO LOOK UP
;    67          0067  1 !                                           USER-DEFINED SYMBOLS (18).
;    68          0068  1 !  04   25-APR-78        K.D. MORSE          CONVERT TO NATIVE COMPILER.
;    69          0069  1 !  05   18-MAY-78        K.D. MORSE          NO CHANGES FOR VERS 19.
;    70          0070  1 !  06   06-JUN-78        K.D. MORSE          REMOVE PAT$VAL_TO_DEF AND ADD
;    71          0071  1 !                                           CODE TO PAT$FIND_VAL TO FIND
;    72          0072  1 !                                           CLOSEST VALUE.
;    73          0073  1 !  07   07-JUN-78        K.D. MORSE          ADD PAT$ADD_LABELS.
;    74          0074  1 !  08   13-JUN-78        K.D. MORSE          ADD FAO COUNTS TO SIGNALS.
;    75          0075  1 !
;    76          0076  1 !--
```


65

65
58

```
;    78          0077  1 FORWARD ROUTINE
;    79          0078  1           PAT$FIND_VAL,                          ! Lookup symbol given value
;    80          0079  1           PAT$DEFINE_SYM : NOVALUE,              ! Stores a user-defined symbol in the symbol
;    81          0080  1           PAT$FIND_SYM,                          ! Finds a symbol in symbol table
;    82          0081  1           PAT$ADD_LABELS : NOVALUE;              ! Adds labels from one symbol list to user-d
;    83          0082  1
;    84          0083  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;    85          0084  1 REQUIRE 'SRC$:VXSMAC.REQ';
;    86          0149  1 REQUIRE 'SRC$:PATPCT.REQ';
;    87          0189  1 REQUIRE 'SRC$:PATGEN.REQ';
;    88          0411  1 REQUIRE 'LIB$:PATDEF.REQ';                        ! Defines literals
;    89          0465  1 REQUIRE 'LIB$:PATMSG.REQ';
;    90          0639  1 REQUIRE 'SRC$:BSTRUC.REQ';
;    91          0715  1 REQUIRE 'SRC$:DLLNAM.REQ';
;    92          0773  1 REQUIRE 'SRC$:SYSSER.REQ';
```

```
;  R0805  1     SWITCHES LIST (SOURCE);
;  R0806  1
;  R0807  1     EXTERNAL ROUTINE
;  R0808  1              PAT$fao_out;           ! formats a line and outputs to the terminal
;  R0809  1
```

```
;    93          0855  1
;    94          0856  1 EXTERNAL ROUTINE
;    95          0857  1         PAT$FREERELEASE,                    ! Deallocates a block of free storage
;    96          0858  1         PAT$FREEZ;                          ! Allocates a block of free storage
;    97          0859  1
;    98          0860  1 EXTERNAL
;    99          0861  1         PAT$GL_SYMHEAD,                     ! Pointer to listhead entry of user-defined
;   100          0862  1         PAT$GL_SYMTBPTR;                    ! Symbol table listhead
;   101          0863  1
;   102          0864  1 BUILTIN
;   103          0865  1         INSQUE;
;   104          0866  1
```

```
;  106     0867  1 GLOBAL ROUTINE PAT$FIND_VAL (VALUE, MATCH_FLAG) =
;  107     0868  1
;  108     0869  1 !++
;  109     0870  1 ! Functional description:
;  110     0871  1 !
;  111     0872  1 !       Searches (in a binary manner) through the doubly-linked symbol
;  112     0873  1 !       table for the name of a symbol which matches the given value.
;  113     0874  1 !       If such a symbol is found, the address of the entry is
;  114     0875  1 !       returned.  Otherwise 0 is returned.
;  115     0876  1 !
;  116     0877  1 !       If MATCH_FLAG is TRUE, an exact match must be found.
;  117     0878  1 !       Otherwise we return a pointer to the closest lower symbol.
;  118     0879  1 !
;  119     0880  1 ! Calling sequence:
;  120     0881  1 !
;  121     0882  1 !       CALLS #2, PAT$FIND_VAL
;  122     0883  1 !
;  123     0884  1 ! Inputs:
;  124     0885  1 !
;  125     0886  1 !       VALUE           -the key we are to search for
;  126     0887  1 !       MATCH_FLAG      -TRUE => insist on an exact match,
;  127     0888  1 !                        FALSE => the closest one will do.
;  128     0889  1 !
;  129     0890  1 ! Implicit inputs:
;  130     0891  1 !
;  131     0892  1 !       PAT$GL_SYMTBPTR, pointer to the header link in the current symbol table.
;  132     0893  1 !
;  133     0894  1 ! Outputs:
;  134     0895  1 !
;  135     0896  1 !       The address of the symbol entry, or zero, if the name is not found.
;  136     0897  1 !
;  137     0898  1 ! Implicit outputs:
;  138     0899  1 !
;  139     0900  1 !       none
;  140     0901  1 !
;  141     0902  1 ! Routine value:
;  142     0903  1 !
;  143     0904  1 !       an address or zero
;  144     0905  1 !
;  145     0906  1 ! Side effects:
;  146     0907  1 !
;  147     0908  1 !       none
;  148     0909  1 !--
;  149     0910  1
;  150     0911  2 BEGIN
;  151     0912  2
;  152     0913  2 LOCAL
;  153     0914  2         RETURN_PTR,                             ! Pointer to closest matched entry
;  154     0915  2         POINTER;                                ! Pointer to current symbol entry
;  155     0916  2
;  156     0917  2 !++
;  157     0918  2 ! Assume there is no closest match (RETURN_PTR = 0).
;  158     0919  2 !--
;  159     0920  2 RETURN_PTR = 0;
;  160     0921  2
;  161     0922  2 !++
;  162     0923  2 ! Set pointer to the first entry in the table.  Loop, searching the table
```

```
;  163          0924  2 ! until an exact match is found or there is no more table.
;  164          0925  2 !--
;  165          0926  2 POINTER = .DLL_RLINK (.PAT$GL_SYMTBPTR);
;  166          0927  2 WHILE .POINTER NEQ .PAT$GL_SYMTBPTR
;  167          0928  2 DO
;  168          0929  3         BEGIN
;  169          0930  3         !++
;  170          0931  3         ! Look for exact match.
;  171          0932  3         !--
;  172          0933  4         IF (.SYM_VALUE(.POINTER) EQLU .VALUE)
;  173          0934  3         THEN
;  174          0935  3                 RETURN .POINTER;
;  175          0936  3
;  176          0937  3         !++
;  177          0938  3         ! Check if closest match is desired.  If so, check that current entry
;  178          0939  3         ! is less than the value sought and if it is closer than RETURN_PTR.
;  179          0940  3         !--
;  180          0941  4         IF (NOT .MATCH_FLAG)
;  181          0942  3         THEN
;  182          0943  4                 IF (.SYM_VALUE(.POINTER) LEQU .VALUE)
;  183          0944  3                 THEN
;  184          0945  4                         BEGIN
;  185          0946  5                         IF (.RETURN_PTR EQLA 0)
;  186          0947  4                         THEN
;  187          0948  4                                 RETURN_PTR = .POINTER
;  188          0949  4                         ELSE
;  189          0950  5                                 IF (.SYM_VALUE(.RETURN_PTR) LEQU .SYM_VALUE(.POINTER))
;  190          0951  4                                 THEN
;  191          0952  4                                         RETURN_PTR = .POINTER;
;  192          0953  3                         END;
;  193          0954  3
;  194          0955  3         !++
;  195          0956  3         ! Go back and look at the next element on the list.
;  196          0957  3         !--
;  197          0958  3         POINTER = .DLL_RLINK(.POINTER);
;  198          0959  2         END;
;  199          0960  2 RETURN(.RETURN_PTR);
;  200          0961  1 END;


                                                        .TITLE  PATSYM
                                                        .IDENT  \V04-000\

                                                        .EXTRN  PAT$FAO_OUT, PAT$FREERELEASE
                                                        .EXTRN  PAT$FREEZ, PAT$GL_SYMHEAD
                                                        .EXTRN  PAT$GL_SYMTBPTR
                                                        .WEAK   ACCESS_CHECK

                                                        .PSECT  _PAT$CODE,NOWRT,2

                                        0004 00000      .ENTRY  PAT$FIND_VAL, Save R2                           ; 0867
                                    51  D4 00002       CLRL    RETURN_PTR                                      ; 0920
                          50 00000000G  FF  D0 00004    MOVL    @PAT$GL_SYMTBPTR, POINTER                       ; 0926
                                52  08  AC  D2 0000B    MCOML   MATCH_FLAG, R2                                  ; 0941
                    00000000G   EF      50  D1 0000F 1$:  CMPL    POINTER, PAT$GL_SYMTBPTR                      ; 0927
                                        1F  13 00016      BEQL    4$                                            ;
                          04    AC  08  A0  D1 00018      CMPL    8(POINTER), VALUE                             ; 0933
```

```
                                   1B   13 0001D            BEQL     5$                                                  ;
                 10                52   E9 0001F            BLBC     R2, 3$                                              ; 0941
                                   0E   1A 00022            BGTRU    3$                                                  ; 0943
                                   51   D5 00024            TSTL     RETURN_PTR                                          ; 0946
                                   07   13 00026            BEQL     2$                                                  ;
          08     A0        08      A1   D1 00028            CMPL     8(RETURN_PTR), 8(POINTER)                           ; 0950
                                   03   1A 0002D            BGTRU    3$                                                  ;
                 51                50   D0 0002F  2$:       MOVL     POINTER, RETURN_PTR                                 ; 0952
                 50                60   D0 00032  3$:       MOVL     (POINTER), POINTER                                  ; 0958
                                   D8   11 00035            BRB      1$                                                  ; 0927
                 50                51   D0 00037  4$:       MOVL     RETURN_PTR, R0                                      ; 0960
                                        04 0003A  5$:       RET                                                          ; 0961

; Routine Size:  59 bytes,    Routine Base:  _PAT$CODE + 0000
```

```
;  202          0962  1 GLOBAL ROUTINE PAT$DEFINE_SYM (STRING_DESC, VALUE, MSG_FLAG) : NOVALUE =
;  203          0963  1
;  204          0964  1 !++
;  205          0965  1 ! Functional description:
;  206          0966  1 !
;  207          0967  1 !       Stores a user-defined symbol in the symbol table, which is a doubly
;  208          0968  1 !       linked list.  First searches (in a binary fashion) the table to see
;  209          0969  1 !       whether the symbol exists.  If it does not, then the symbol is
;  210          0970  1 !       inserted, except if no free storage is left, in which case another
;  211          0971  1 !       error message is reported from PAT$FREEZ.
;  212          0972  1 !
;  213          0973  1 ! Calling sequence:
;  214          0974  1 !
;  215          0975  1 !       CALLS #2, PAT$DEFINE_SYM
;  216          0976  1 !
;  217          0977  1 ! Inputs:
;  218          0978  1 !
;  219          0979  1 !       STRING_DESC     - a string descriptor describing the string
;  220          0980  1 !                         representing the symbol.
;  221          0981  1 !       VALUE           - the value to be stored as the equivalent of the
;  222          0982  1 !                         symbol.
;  223          0983  1 !       MSG_FLAG        - Indicator whether or not to print a define message,
;  224          0984  1 !                         (TRUE=print message, FALSE=don't print message)
;  225          0985  1 !
;  226          0986  1 ! Implicit inputs:
;  227          0987  1 !
;  228          0988  1 !       PAT$GL_SYMTBPTR, pointer to the header link in the current symbol table.
;  229          0989  1 !
;  230          0990  1 ! Implicit outputs:
;  231          0991  1 !
;  232          0992  1 !       none
;  233          0993  1 !
;  234          0994  1 ! Routine value:
;  235          0995  1 !
;  236          0996  1 !       novalue
;  237          0997  1 !
;  238          0998  1 ! Side effects:
;  239          0999  1 !
;  240          1000  1 !       The symbol is inserted into the symbol table.  The links of
;  241          1001  1 !       the table are appropriately adjusted.
;  242          1002  1 !
;  243          1003  1 !--
;  244          1004  1
;  245          1005  2 BEGIN
;  246          1006  2
;  247          1007  2 MAP
;  248          1008  2         STRING_DESC : REF BLOCK [, BYTE];
;  249          1009  2
;  250          1010  2 LOCAL
;  251          1011  2         POINTER;
;  252          1012  2
;  253          1013  2 POINTER = PAT$FIND_SYM (.STRING_DESC);
;  254          1014  3 IF (.POINTER EQL 0)
;  255          1015  2 THEN
;  256          1016  3         BEGIN
;  257          1017  3         !++
;  258          1018  3         ! Symbol did not exist. Insert it into the symbol table.  First allocate
```

```
;  259          1019  3          ! a block of dynamic storage to hold it.
;  260          1020             !--
;  261          1021             POINTER = PAT$FREEZ ((.STRING_DESC [DSC$W_LENGTH] + 1 + 3) / 4 + OVERHEAD_SYM - 1);
;  262          1022
;  263          1023             !++
;  264          1024             ! Space allocated. Insert the new link. Then write the symbol name and
;  265          1025             ! value into the new symbol table entry.
;  266          1026             !--
;  267          1027             INSQUE (.POINTER, .PAT$GL_SYMTBPTR);
;  268          1028             CH$MOVE (.STRING_DESC [DSC$W_LENGTH], CH$PTR (.STRING_DESC [DSC$A_POINTER]),
;  269          1029                      CH$PTR (SYM_NAME (.POINTER)));
;  270          1030             SYM_CHCOUNT (.POINTER) = .STRING_DESC [DSC$W_LENGTH];
;  271          1031             SYM_VALUE (.POINTER) = .VALUE;
;  272          1032             IF (.MSG_FLAG)
;  273          1033             THEN
;  274        P 1034                     $FAO_TT_OUT('symbol "!AD" defined as !XL',
;  275        P 1035                                 .STRING_DESC[DSC$W_LENGTH],
;  276          1036                                 CH$PTR(.STRING_DESC[DSC$A_POINTER]), .VALUE);
;  277          1037             END
;  278          1038  2  ELSE
;  279          1039             BEGIN
;  280          1040             !++
;  281          1041             ! Symbol already was defined. Just reset the value of
;  282          1042             ! the symbol stored in the symbol table.
;  283          1043             !--
;  284          1044             IF (.MSG_FLAG)
;  285          1045             THEN
;  286        P 1046                     $FAO_TT_OUT('symbol "!AD" redefined from !XL to !XL',
;  287        P 1047                                 .STRING_DESC[DSC$W_LENGTH],
;  288        P 1048                                 CH$PTR(.STRING_DESC[DSC$A_POINTER]),
;  289          1049                                 .SYM_VALUE(.POINTER), .VALUE);
;  290          1050             SYM_VALUE (.POINTER) = .VALUE;
;  291          1051  2          END;
;  292          1052  1 END;


                                                                                .PSECT  _PAT$PLIT,NOWRT,NOEXE,0

                                                                        1B  00000 P.AAA:  .BYTE   27
65  64  20  22  44  41  21  22  20  6C  6F  62  6D  79  73  00001               .ASCII  \symbol "!AD" defined as !XL\                   ;
            4C  58  21  20  73  61  20  64  65  6E  69  66  00010                                                                       ;
                                                        26  0001C P.AAB:  .BYTE   38                                                ;
65  72  20  22  44  41  21  22  20  6C  6F  62  6D  79  73  0001D               .ASCII  \symbol "!AD" redefined from !XL to !XL\        ;
58  21  20  6D  6F  72  66  20  64  65  6E  69  66  65  64  0002C                                                                       ;
                        4C  58  21  20  6F  74  20  4C  0003B                                                                       ;


                                                                                .PSECT  _PAT$CODE,NOWRT,2

                                                                    01FC  00000         .ENTRY  PAT$DEFINE_SYM, Save R2,R3,R4,R5,R6,R7,R8   ; 0962
                                                 58 00000000G   EF   9E  00002         MOVAB   PAT$FAO_OUT, R8                             ;
                                                 52       04    AC   D0  00009         MOVL    STRING_DESC, R2                             ; 1013
                                                               52   DD  0000D         PUSHL   R2                                          ;
                                       00000000V EF            01   FB  0000F         CALLS   #1, PAT$FIND_SYM                            ;
                                                 57            50   D0  00016         MOVL    R0, POINTER                                 ;
```

```
                                                47  12 00019          BNEQ    1$                                      ; 1014
                                        50      62  3C 0001B          MOVZWL  (R2), R0                                ; 1021
                                        50      04  C0 0001E          ADDL2   #4, R0                                  ;
                                        50      04  C6 00021          DIVL2   #4, R0                                  ;
                                            03  A0  9F 00024          PUSHAB  3(R0)                                   ;
                                00000000G   EF      01  FB 00027      CALLS   #1, PAT$FREEZ                           ;
                                        57      50  D0 0002E          MOVL    R0, POINTER                             ;
                                00000000G   FF      67  0E 00031      INSQUE  (POINTER), @PAT$GL_SYMTBPTR             ; 1027
                                        56      04  AC  D0 00038      MOVL    STRING_DESC, R6                         ; 1028
                    0D  A7          04      B6      66  28 0003C      MOVC3   (R6), @4(R6), 13(POINTER)               ; 1029
                                    0C      A7      66  90 00042      MOVB    (R6), 12(POINTER)                       ; 1030
                                    08      A7      08  AC  D0 00046  MOVL    VALUE, 8(POINTER)                       ; 1031
                                    31          0C  AC  E9 0004B      BLBC    MSG_FLAG, 3$                            ; 1032
                                                08  AC  DD 0004F      PUSHL   VALUE                                   ; 1036
                                                04  A6  DD 00052      PUSHL   4(R6)                                   ;
                                        7E          66  3C 00055      MOVZWL  (R6), -(SP)                             ;
                                    00000000'   EF      9F 00058      PUSHAB  P.AAA                                   ;
                                        68              04  FB 0005E  CALLS   #4, PAT$FAO_OUT                         ;
                                                        04  00061     RET                                             ; 1014
                                        15          0C  AC  E9 00062  1$:     BLBC    MSG_FLAG, 2$                  ; 1044
                                                08  AC  DD 00066      PUSHL   VALUE                                   ; 1049
                                                08  A7  DD 00069      PUSHL   8(POINTER)                              ;
                                                04  A2  DD 0006C      PUSHL   4(R2)                                   ;
                                        7E          62  3C 0006F      MOVZWL  (R2), -(SP)                             ;
                                    00000000'   EF      9F 00072      PUSHAB  P.AAB                                   ;
                                        68              05  FB 00078  CALLS   #5, PAT$FAO_OUT                         ;
                                    08  A7          08  AC  D0 0007B  2$:     MOVL    VALUE, 8(POINTER)               ; 1050
                                                        04  00080  3$:     RET                                         ; 1052

; Routine Size:  129 bytes,    Routine Base:  _PAT$CODE + 003B
```

```
;  294          1053  1 GLOBAL ROUTINE PAT$FIND_SYM (STRING_DESC) =
;  295          1054  1
;  296          1055  1 !++
;  297          1056  1 ! Functional description:
;  298          1057  1 !
;  299          1058  1 !       Searches (in a binary manner) through the doubly-linked symbol
;  300          1059  1 !       table for a symbol name. If it is found, the address of the entry
;  301          1060  1 !       is returned. Else a zero is returned.
;  302          1061  1 !
;  303          1062  1 ! Calling sequence:
;  304          1063  1 !
;  305          1064  1 !       CALLS #1, PAT$FIND_SYM
;  306          1065  1 !
;  307          1066  1 ! Inputs:
;  308          1067  1 !
;  309          1068  1 !       STRING_DESC     - the string descriptor of the symbol to find
;  310          1069  1 !
;  311          1070  1 ! Implicit inputs:
;  312          1071  1 !
;  313          1072  1 !       PAT$GL_SYMTBPTR, pointer to the header link in the current symbol table.
;  314          1073  1 !
;  315          1074  1 ! Outputs:
;  316          1075  1 !
;  317          1076  1 !       The address of the symbol entry, or zero, if the name is not found.
;  318          1077  1 !
;  319          1078  1 ! Implicit outputs:
;  320          1079  1 !
;  321          1080  1 !       none
;  322          1081  1 !
;  323          1082  1 ! Routine value:
;  324          1083  1 !
;  325          1084  1 !       an address or zero
;  326          1085  1 !
;  327          1086  1 ! Side effects:
;  328          1087  1 !
;  329          1088  1 !       none
;  330          1089  1 !
;  331          1090  1 !--
;  332          1091  1
;  333          1092  2 BEGIN
;  334          1093  2
;  335          1094  2 MAP
;  336          1095  2         STRING_DESC : REF BLOCK [, BYTE];
;  337          1096  2
;  338          1097  2 LOCAL
;  339          1098  2         POINTER;
;  340          1099  2
;  341          1100  2 POINTER = .DLL_RLINK (.PAT$GL_SYMTBPTR);
;  342          1101  3 WHILE (.POINTER NEQ .PAT$GL_SYMTBPTR)
;  343          1102  2 DO
;  344          1103  3         BEGIN
;  345          1104  3         IF CH$EQL (.STRING_DESC [DSC$W_LENGTH], CH$PTR (.STRING_DESC [DSC$A_POINTER], 0),
;  346          1105  3                 .SYM_CHCOUNT (.POINTER), CH$PTR (SYM_NAME (.POINTER), 0))
;  347          1106  3         THEN RETURN .POINTER
;  348          1107  3         ELSE POINTER = .DLL_RLINK (.POINTER);
;  349          1108  2         END;
;  350          1109  2 RETURN 0
```

```
;  351          1110  1 END;


                                                003C 00000          .ENTRY  PAT$FIND_SYM, Save R2,R3,R4,R5                  ; 1053
                                    54 00000000G    FF   D0 00002       MOVL    @PAT$GL_SYMTBPTR, POINTER                   ; 1100
                                    55          04  AC   D0 00009       MOVL    STRING_DESC, R5                             ; 1104
                          00000000G EF              54   D1 0000D 1$:     CMPL    POINTER, PAT$GL_SYMTBPTR                    ; 1101
                                                    18   13 00014       BEQL    3$
                                    50          0C  A4   9A 00016       MOVZBL  12(POINTER), R0                             ; 1105
          50            00        04  B5          04   BC   2D 0001A       CMPC5   @STRING_DESC, @4(R5), #0, R0, 13(POINTER)
                                                0D  A4       00021
                                                    04   12 00023       BNEQ    2$
                                    50              54   D0 00025       MOVL    POINTER, R0                                 ; 1106
                                                    04   00028       RET
                                    54              64   D0 00029 2$:     MOVL    (POINTER), POINTER                          ; 1107
                                                    DF   11 0002C       BRB     1$                                          ; 1101
                                                    50   D4 0002E 3$:     CLRL    R0                                          ; 1109
                                                         04 00030       RET                                                 ; 1110

; Routine Size:  49 bytes,    Routine Base:  _PAT$CODE + 00BC
```

```
;  353     1111  1 GLOBAL ROUTINE PAT$ADD_LABELS (SYM_LISTHEAD) : NOVALUE =
;  354     1112  1
;  355     1113  1 !++
;  356     1114  1 ! Functional description:
;  357     1115  1 !
;  358     1116  1 !       This routine merges a label symbol table into the user-defined
;  359     1117  1 !       symbol table.  The label symbol table entries are identical in
;  360     1118  1 !       format to the user-defined symbol table.  Therefore, for new
;  361     1119  1 !       symbols, the entry can be linked into the user table without
;  362     1120  1 !       alteration.  Redefined symbols merely alter the user table entry
;  363     1121  1 !       and are then released to free storage.
;  364     1122  1 !
;  365     1123  1 ! Calling sequence:
;  366     1124  1 !
;  367     1125  1 !       CALLS #1, PAT$ADD_LABELS
;  368     1126  1 !
;  369     1127  1 ! Inputs:
;  370     1128  1 !
;  371     1129  1 !       SYM_LISTHEAD    - Address of the pointer to the label symbol table
;  372     1130  1 !                         to be added
;  373     1131  1 !
;  374     1132  1 ! Implicit inputs:
;  375     1133  1 !
;  376     1134  1 !       PAT$GL_SYMTBPTR, pointer to the header link in the current symbol table.
;  377     1135  1 !
;  378     1136  1 ! Outputs:
;  379     1137  1 !
;  380     1138  1 !       none
;  381     1139  1 !
;  382     1140  1 ! Implicit outputs:
;  383     1141  1 !
;  384     1142  1 !       none
;  385     1143  1 !
;  386     1144  1 ! Routine value:
;  387     1145  1 !
;  388     1146  1 !       none
;  389     1147  1 !
;  390     1148  1 ! Side effects:
;  391     1149  1 !
;  392     1150  1 !       The label symbol table contains only its listhead entry and all
;  393     1151  1 !       labels are now in the user-defined symbol table.  The current symbol
;  394     1152  1 !       table pointer, PAT$GL_SYMTBPTR, is reset to the user-defined symbol table.
;  395     1153  1 !
;  396     1154  1 !--
;  397     1155  1
;  398     1156  1
;  399     1157  2 BEGIN
;  400     1158  2
;  401     1159  2 MAP
;  402     1160  2         SYM_LISTHEAD : REF VECTOR[,LONG];                       ! Pointer to listhead entry of label table
;  403     1161  2
;  404     1162  2 LOCAL
;  405     1163  2         STRING_DESC : BLOCK[8,BYTE],                            ! String descriptor for current symbol
;  406     1164  2         SYMBOL_PTR,                                             ! Pointer to symbol entry in user-defined ta
;  407     1165  2         NEXT_PTR,                                               ! Pointer to next label entry
;  408     1166  2         POINTER;
;  409     1167  2
```

```
;  410          1168  2 PAT$GL_SYMTBPTR = .PAT$GL_SYMHEAD;
;  411          1169  2 WHILE (POINTER=.DLL_RLINK(.SYM_LISTHEAD[0])) NEQA .SYM_LISTHEAD[0]
;  412          1170  2 DO
;  413          1171  3         BEGIN
;  414          1172  3         NEXT_PTR = .DLL_RLINK(.POINTER);
;  415          1173  3         DLL_LLINK(.NEXT_PTR) = .DLL_LLINK(.POINTER);
;  416          1174  3         DLL_RLINK(.SYM_LISTHEAD[0]) = .DLL_RLINK(.POINTER);
;  417          1175  3         STRING_DESC[DSC$W_LENGTH] = .SYM_CHCOUNT(.POINTER);
;  418          1176  3         STRING_DESC[DSC$A_POINTER] = SYM_CSTRING(.POINTER) + 1;
;  419          1177  3         SYMBOL_PTR = PAT$FIND_SYM(STRING_DESC);
;  420          1178  4         IF (.SYMBOL_PTR EQLA 0)
;  421          1179  3         THEN
;  422          1180  4                 BEGIN
;  423          1181  4                 !++
;  424          1182  4                 ! This is a new user symbol.  Add the entry to the user-
;  425          1183  4                 ! defined symbol table.
;  426          1184  4                 !--
;  427          1185  4                 DLL_RLINK(.POINTER) = 0;
;  428          1186  4                 DLL_LLINK(.POINTER) = 0;
;  429          1187  4                 INSQUE(.POINTER, .PAT$GL_SYMHEAD);
;  430        P 1188  4                 $FAO_TT_OUT('symbol "!AD" defined as !XL',
;  431        P 1189  4                             .STRING_DESC[DSC$W_LENGTH],
;  432        P 1190  4                             CH$PTR(.STRING_DESC[DSC$A_POINTER]),
;  433          1191  4                             .SYM_VALUE(.POINTER));
;  434          1192  4                 END
;  435          1193  3         ELSE
;  436          1194  4                 BEGIN
;  437          1195  4                 !++
;  438          1196  4                 ! This is a redefinition of a user symbol.  Alter the value
;  439          1197  4                 ! in the user-defined symbol table and then release the
;  440          1198  4                 ! label entry to the free storage list.
;  441          1199  4                 !--
;  442        P 1200  4                 $FAO_TT_OUT('symbol "!AD" redefined from !XL to !XL',
;  443        P 1201  4                         .STRING_DESC[DSC$W_LENGTH], CH$PTR(.STRING_DESC[DSC$A_POINTER]),
;  444          1202  4                         .SYM_VALUE(.SYMBOL_PTR), .SYM_VALUE(.POINTER));
;  445          1203  4                 SYM_VALUE(.SYMBOL_PTR) = .SYM_VALUE(.POINTER);
;  446          1204  5                 PAT$FREERELEASE(.POINTER, ((.STRING_DESC[DSC$W_LENGTH] + 1 + 3)/
;  447          1205  4                                            4 + OVERHEAD_SYM - 1));
;  448          1206  3                 END;
;  449          1207  3
;  450          1208  2         END;
;  451          1209  1 END;


                                                                .PSECT  _PAT$PLIT,NOWRT,NOEXE,0

                                                        1B  00043 P.AAC:  .BYTE   27
65  64  20  22  44  41  21  22  20  6C  6F  62  6D  79  73  00044         .ASCII  \symbol "!AD" defined as !XL\                    ;
                4C  58  21  20  73  61  20  64  65  6E  69  66  00053
                                                        26  0005F P.AAD:  .BYTE   38
65  72  20  22  44  41  21  22  20  6C  6F  62  6D  79  73  00060         .ASCII  \symbol "!AD" redefined from !XL to !XL\         ;
58  21  20  6D  6F  72  66  20  64  65  6E  69  66  65  64  0006F
                        4C  58  21  20  6F  74  20  4C  0007E                                                                    ;
```

```
                                                .PSECT  _PAT$CODE,NOWRT,2

                                        003C 00000      .ENTRY  PAT$ADD_LABELS, Save R2,R3,R4,R5                ; 1111
                55 00000000G    EF  9E 00002            MOVAB   PAT$FAO_OUT, R5                                 ;
                5E              08  C2 00009            SUBL2   #8, SP                                          ;
00000000G       EF 00000000G    EF  D0 0000C            MOVL    PAT$GL_SYMHEAD, PAT$GL_SYMTBPTR                 ; 1168
                50          04  BC  D0 00017 1$:         MOVL    @SYM_LISTHEAD, R0                               ; 1169
                52              60  D0 0001B            MOVL    (R0), POINTER                                   ;
                50              52  D1 0001E            CMPL    POINTER, R0                                     ;
                                6F  13 00021            BEQL    3$                                              ;
                54              62  D0 00023            MOVL    (POINTER), NEXT_PTR                             ; 1172
            04  A4          04 A2  D0 00026            MOVL    4(POINTER), 4(NEXT_PTR)                         ; 1173
                60              62  D0 0002B            MOVL    (POINTER), (R0)                                 ; 1174
                6E          0C  A2  9B 0002E            MOVZBW  12(POINTER), STRING_DESC                        ; 1175
            04  AE          0D  A2  9E 00032            MOVAB   13(R2), STRING_DESC+4                           ; 1176
                5E              DD  00037               PUSHL   SP                                              ; 1177
            92  AF              01  FB 00039            CALLS   #1, PAT$FIND_SYM                                ;
                53              50  D0 0003D            MOVL    R0, SYMBOL_PTR                                  ;
                                1E  12 00040            BNEQ    2$                                              ; 1178
                                62  7C 00042            CLRQ    (POINTER)                                       ; 1185
00000000G       FF              62  0E 00044            INSQUE  (POINTER), @PAT$GL_SYMHEAD                      ; 1187
                            08  A2  DD 0004B            PUSHL   8(POINTER)                                      ; 1191
                            08  AE  DD 0004E            PUSHL   STRING_DESC+4                                   ;
                7E          08  AE  3C 00051            MOVZWL  STRING_DESC, -(SP)                              ;
                    00000000'   EF  9F 00055            PUSHAB  P.AAC                                           ;
                65              04  FB 0005B            CALLS   #4, PAT$FAO_OUT                                 ;
                                B7  11 0005E            BRB     1$                                              ; 1178
                            08  A2  DD 00060 2$:        PUSHL   8(POINTER)                                      ; 1202
                            08  A3  DD 00063            PUSHL   8(SYMBOL_PTR)                                   ;
                            0C  AE  DD 00066            PUSHL   STRING_DESC+4                                   ;
                7E          0C  AE  3C 00069            MOVZWL  STRING_DESC, -(SP)                              ;
                    00000000'   EF  9F 0006D            PUSHAB  P.AAD                                           ;
                65              05  FB 00073            CALLS   #5, PAT$FAO_OUT                                 ;
            08  A3          08 A2  D0 00076            MOVL    8(POINTER), 8(SYMBOL_PTR)                       ; 1203
                50              6E  3C 0007B            MOVZWL  STRING_DESC, R0                                 ; 1204
                50              04  C0 0007E            ADDL2   #4, R0                                          ;
                50              04  C6 00081            DIVL2   #4, R0                                          ;
                            03  A0  9F 00084            PUSHAB  3(R0)                                           ;
                                52  DD 00087            PUSHL   POINTER                                         ;
                00000000G   EF              02  FB 00089        CALLS   #2, PAT$FREERELEASE                     ;
                                            85  11 00090        BRB     1$                                      ; 1169
                                            04  00092 3$:     RET                                             ; 1209

; Routine Size:  147 bytes,    Routine Base:  _PAT$CODE + 00ED
```

```
;  453          1210  1 END
;  454          1211  0 ELUDOM



;                                   PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;
;  _PAT$CODE                       384  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)
;  _PAT$PLIT                       134  NOVEC,NOWRT,  RD ,NOEXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(0)


;                                 Library Statistics
;
;                                        --------- Symbols ---------      Pages       Processing
;       File                             Total    Loaded   Percent      Mapped       Time
;
;  _$255$DUA28:[SYSLIB]LIB.L32;1          18619         6         0        1000         00:01.8



;                                       COMMAND QUALIFIERS

;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/VARIANT:1/LIS=LIS$:PATSYM/OBJ=OBJ$:PATSYM MSRC$:PATSYM/UPDATE=(ENH$:PATSYM)

; Size:         384 code + 134 data bytes
; Run Time:       00:21.0
; Elapsed Time:   01:14.8
; Lines/CPU Min:   3468
; Lexemes/CPU-Min: 45579
; Memory Used:  133 pages
; Compilation Complete
```

0304 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

PHONE

PATSYM
LIS

PHONE
MAP

BASICCMDS
LIS

PATSTO
LIS

PATVEC
LIS

PHONEREQ
REQ

PATWRT
LIS

FILECMDS
LIS